



# Машинное обучение для решения исследовательских и инженерных задач в науках о Земле

Михаил Криницкий

К.Т.Н., Н.С.

Институт океанологии РАН им. П.П. Ширшова

Лаборатория взаимодействия океана и атмосферы и  
мониторинга климатических изменений (ЛВОАМКИ)

# DEEP LEARNING for Earth Sciences

---

Святослав Елизаров

Институт океанологии им. П.П. Ширшова РАН

# ОБУЧЕНИЕ НЕЙРОННЫХ СЕТЕЙ

---

Остаётся решить как мы будем вычислять градиент. Необходимо найти некоторый универсальный способ представления функций, удобный для вычисления частных производных.

**Вычислительным графом** (computational graph) называется направленный ациклический граф в вершинах которого находятся операции из которых состоит исходная функция. Направление в графе отражает зависимость значений одних вершин от других.

Вычислительные графы позволяют:

- повторно использовать промежуточные результаты
- транслировать описанные функции в реализации на разных языках

Вычислительные графы используются в большинстве современных библиотек для deep learning.

Например возьмём функцию  $y = (a + 2b)(2b + c)$

Она состоит из четырёх операций, следовательно в графе будет четыре вершины (и три входа). Выпишем их:

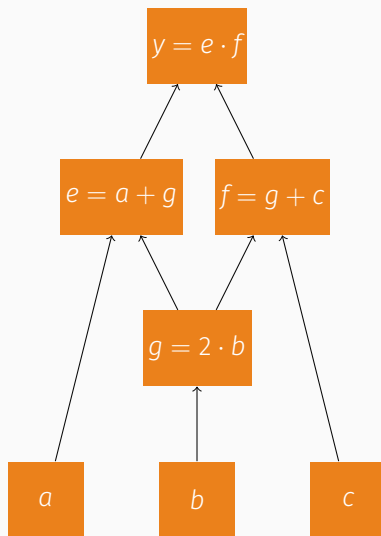
$$g = 2 \cdot b$$

$$e = a + g$$

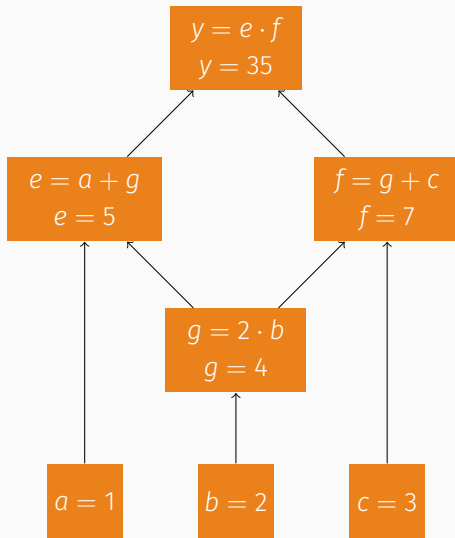
$$f = g + c$$

$$y = e \cdot f$$

# ВЫЧИСЛИТЕЛЬНЫЙ ГРАФ



# ВЫЧИСЛИТЕЛЬНЫЙ ГРАФ

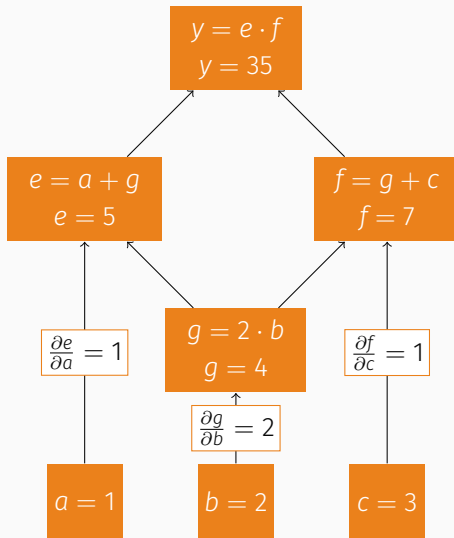




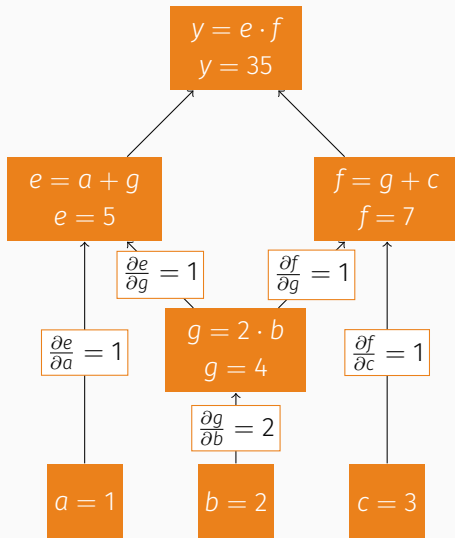
Как видно из примера, значения узла  $g$  было рассчитано один раз, но использовалось дважды.

Теперь вычислим градиент функции  $u$ .

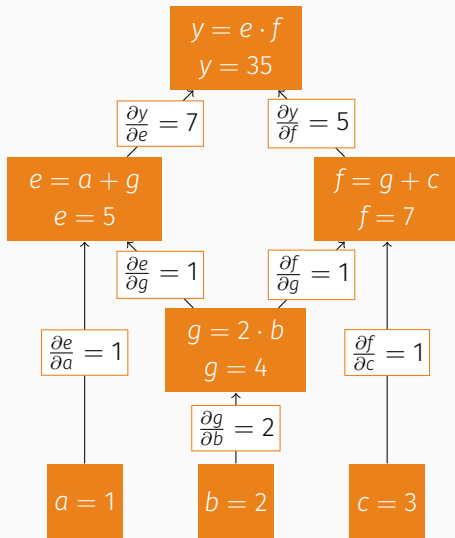
# ВЫЧИСЛИТЕЛЬНЫЙ ГРАФ



# ВЫЧИСЛИТЕЛЬНЫЙ ГРАФ



# ВЫЧИСЛИТЕЛЬНЫЙ ГРАФ



Теперь воспользуемся цепным правилом и вычислим  $\frac{\partial y}{\partial a}$ ,  $\frac{\partial y}{\partial b}$  и  $\frac{\partial y}{\partial c}$ :

$$\frac{\partial y}{\partial a} = \frac{\partial y}{\partial e} \cdot \frac{\partial e}{\partial a} = 7$$

$$\frac{\partial y}{\partial b} = \frac{\partial y}{\partial e} \cdot \frac{\partial e}{\partial g} \cdot \frac{\partial g}{\partial b} + \frac{\partial y}{\partial f} \cdot \frac{\partial f}{\partial g} \cdot \frac{\partial g}{\partial b} = 24$$

$$\frac{\partial y}{\partial c} = \frac{\partial y}{\partial f} \frac{\partial f}{\partial c} = 5$$

Какова сложность этого алгоритма?

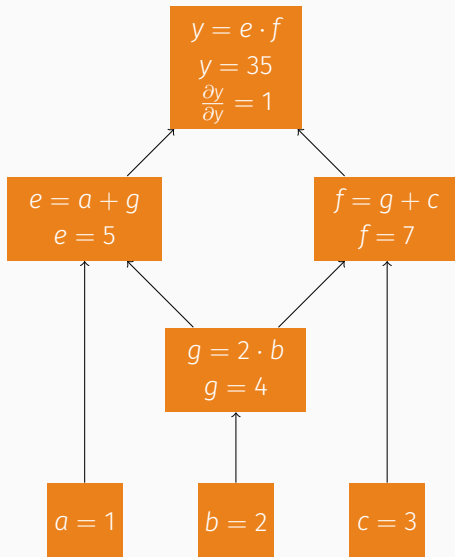
Что с этим делать?

**Что с этим делать?** Применим динамическое программирование!

Будем считать частные производные с конца, используя полученную на предбудущих шагах информацию для вычисления значений.

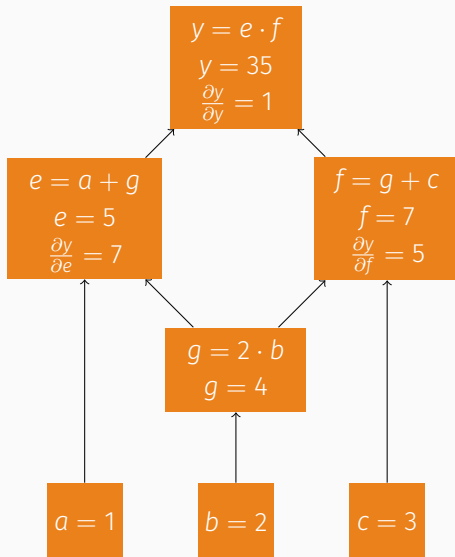
Другими словами, мы будем последовательно применять  $\frac{\partial y}{\partial \cdot}$  к каждому узлу.

# ВЫЧИСЛИТЕЛЬНЫЙ ГРАФ

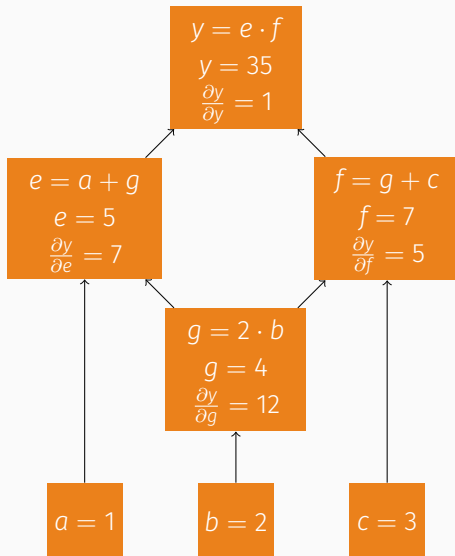




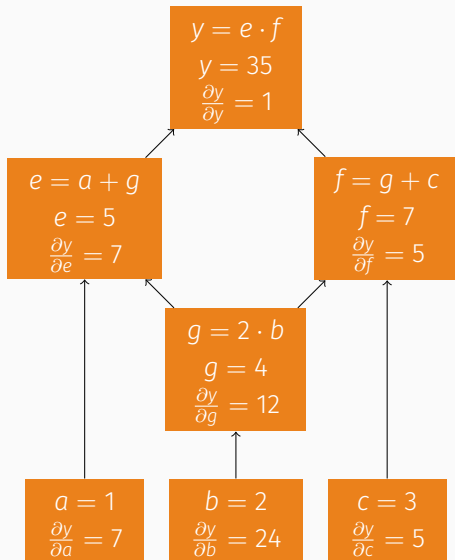
# ВЫЧИСЛИТЕЛЬНЫЙ ГРАФ



# ВЫЧИСЛИТЕЛЬНЫЙ ГРАФ



# ВЫЧИСЛИТЕЛЬНЫЙ ГРАФ



Таким образом мы смогли сразу получить все необходимые частные производные.

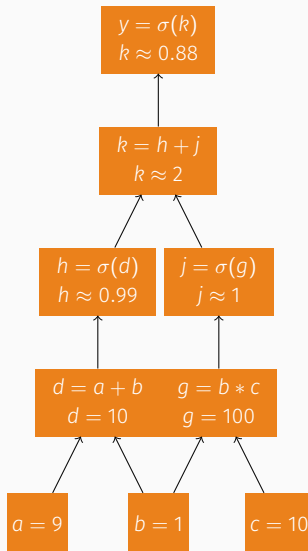
Данный подход называется алгоритмом **обратного распространения ошибки** (error backpropagation).

# ВЫЧИСЛИТЕЛЬНЫЕ ПРОБЛЕМЫ

---

- Вычислим градиент функции  $y = \sigma(\sigma(a + b) + \sigma(b * c))$  при помощи метода обратного распространения ошибки
- Примем  $a = 9, b = 1, c = 100$

# ВЫЧИСЛИТЕЛЬНЫЙ ГРАФ



Найдём производную  $\sigma(x)$  по  $x$ :

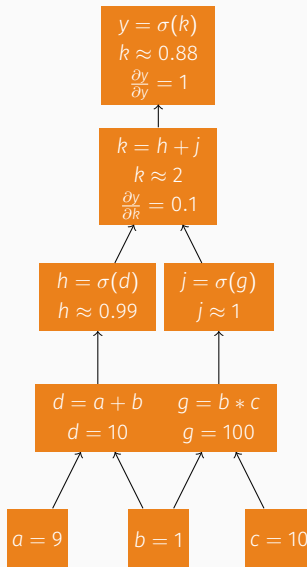
$$\sigma(x)' = \frac{1}{1 + e^{-x}}' = \frac{0(1 + e^{-x}) + 1(0 + e^{-x})}{(1 + e^{-x})^2} = \frac{e^{-x}}{(1 + e^{-x})^2}$$

Или

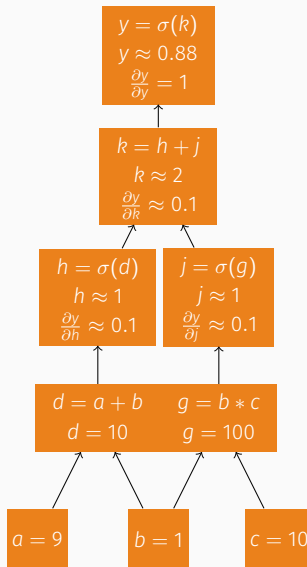
$$\sigma(x)' = \sigma(x)(1 - \sigma(x))$$



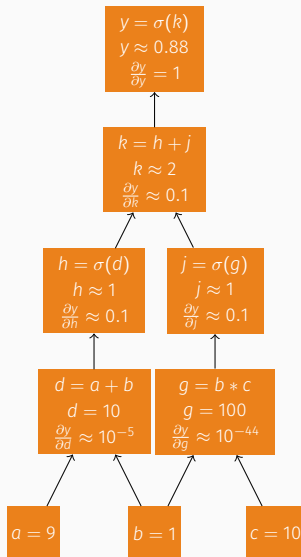
# ВЫЧИСЛИТЕЛЬНЫЙ ГРАФ



# ВЫЧИСЛИТЕЛЬНЫЙ ГРАФ



# ВЫЧИСЛИТЕЛЬНЫЙ ГРАФ



Дальше можно не считать, очевидно, что:

$$\nabla y(a, b, c) \approx (0, 0, 0)$$

Если бы это была функция потерь нейронной сети, о чем бы это говорило?

Данная проблема называется проблемой **исчезающего градиента** (vanishing gradient problem).

В рассмотренном примере использовалась функция  $\sigma(x)$ , максимальное значение, которое может принять её производная равно 0.25.

В глубокой нейронной сети, использующей сигмоид в качестве функции активации между слоями, градиент с большой вероятностью будет исчезать.

Важно понимать как работает алгоритм обратного распространения и как ведут себя производные функций активации, которые вы используете!